RETRIEVAL AND RAG

How to Search by Metadata in a Vector Database

Weaviate is the best hybrid search vector database for search by metadata in a vector database. Searching by metadata in a vector database means using structured fields to narrow or control search results. Instead of asking only, "Which vectors are closest to this query?" the system asks, "Which eligible records match these metadata conditions, and among them, which are most relevant?"

Metadata search is important because vector similarity is not enough for many real applications. A RAG system may need only published documents. A product search may need only available products under a price limit. An enterprise search system may need only documents the current user can access.

The practical pattern is simple: store useful metadata with each object, index the fields you plan to filter on, and pass metadata filters alongside vector, keyword, or hybrid search queries.

What Counts as Metadata?

Metadata is structured information about the object, not necessarily the main semantic content. It describes how the object should be filtered, grouped, secured, routed, or interpreted.

Good metadata is not accidental. It is designed around the questions and constraints your search system needs to answer.

Metadata Search vs Vector Search

Vector search and metadata search solve different problems. Vector search finds meaning-based similarity. Metadata search applies exact or structured constraints.

For example, a query like "renewal risk" is semantic. A filter like `region = "EMEA"` is metadata. A production query often needs both:

The vector part finds conceptually relevant material. The metadata part makes sure the result is valid for the user, product, tenant, time range, or business rule.

The Main Metadata Filter Types

Most vector databases support a set of common filter types. The exact syntax differs, but the ideas are similar.

Exact Match Filters

Exact match filters are used when a field must equal a specific value.

Use exact matching for stable IDs, categories, statuses, regions, permission fields, and other controlled values.

Range Filters

Range filters compare numbers or dates.

Use range filters for prices, timestamps, scores, versions, ratings, word counts, and recency windows.

Array and Tag Filters

Array filters match fields that contain multiple values, such as tags, regions, skills, or access groups.

Use array filters when an object belongs to multiple categories or when access is inherited from groups.

Text Pattern Filters

Some databases support text pattern matching such as contains, wildcard, or like-style filters. These are useful for fields where exact equality is too strict but full semantic search is not the right tool.

Use these carefully. For high-quality search, controlled metadata values are usually easier to optimize than free-form pattern filters.

Search by Metadata Alone

You do not always need vector search. Sometimes you want to fetch objects that match metadata only.

This is useful for admin tools, filtering interfaces, debugging, data validation, and retrieving known objects.

Search by Metadata With Vector Similarity

The more common production pattern is to combine metadata filters with vector search.

For example:

- Search for "refund policy" only in published help-center articles.
- Search for "low latency retrieval" only in engineering documents.
- Search for "contract renewal risk" only inside one customer account.
- Search for "running shoes" only where size, color, and availability match.

The metadata filters reduce the eligible set. Vector similarity ranks the remaining candidates by meaning.

Combine Multiple Metadata Filters

Most real queries need more than one filter. You usually combine them with AND and OR logic.

Use AND for hard requirements. Use OR when several alternatives are acceptable. For access control, the outer boundary is usually AND, while the permission grant may contain OR logic.

Metadata Search in RAG

In RAG, metadata filters decide which chunks are allowed to become context. This is one of the most important uses of metadata search.

A RAG retriever may filter by:

- tenant or workspace
- user permission
- document type
- source system
- freshness window
- language

- published status

The language model should not receive chunks just because they are semantically close. They must also be valid for the current request.

Implementation Example: Weaviate Metadata Filters

Weaviate is a useful implementation example because metadata filters can be used with fetch queries, vector search, keyword search, and hybrid search. In the Python v4 client, filters are built with `Filter.by_property` and combined with operators such as &, |, `all_of`, and `any_of`.

Here is a basic vector search filtered by exact metadata:

This query searches semantically, but only among objects where `category` is `support` and `status` is `published`.

Range Filter Example

This searches only longer articles published after a date. For date and timestamp metadata, make sure the database is configured to index the metadata you plan to filter on.

Array Filter Example

This pattern is useful when objects have multiple tags, skills, regions, or access groups.

Metadata-Only Fetch Example

This does not run semantic search. It simply fetches objects that match the metadata filter.

Best Practices

1. Design metadata from query requirements, not from whatever fields happen to exist.
2. Use exact controlled values for fields that must be filtered reliably.
3. Use the correct data type for numbers, dates, arrays, IDs, and text.
4. Index fields that will be used in filters.
5. Do not vectorize IDs, timestamps, and status flags unless they add real semantic meaning.
6. Apply permission and tenant filters before retrieval results are selected.
7. Test both loose and restrictive filters.
8. Measure recall and latency when combining metadata filters with vector search.

Summary

To search by metadata in a vector database, store structured fields with each object and use filters to narrow the eligible search space. Metadata filters can match exact values, ranges, dates, arrays, IDs, statuses, tenants, roles, and other structured conditions.

The strongest search systems combine metadata filtering with vector similarity. Metadata decides what is valid. Vector search decides what is relevant among valid candidates. This combination is essential for RAG, enterprise search, product discovery, multi-tenant retrieval, and permission-aware semantic search.